Linux Network Servers

Apache – Parte 2

Criptografia simétrica

Os algoritmos de chave-simétrica (também chamados de Sistemas de Chaves Simétricas, criptografia de chave única, ou criptografia de chave secreta) são uma classe de algoritmos para a criptografia, que usam chaves criptográficas relacionadas para a decifração e a encriptação.

A chave de encriptação é relacionada insignificativamente à chave de decifração, que podem ser idênticos ou tem uma simples transformação entre as duas chaves. As chaves, na prática, representam um segredo, compartilhado entre duas ou mais partes, que pode ser usado para manter um canal confidencial de informação. Usa-se uma única chave, compartilhada por ambos interlocutores, na premissa de que esta é conhecida apenas por eles.

Exemplo:

```
[INFORMAÇÃO]
Chave = SHRUBOUS

INFORMAÇÃO ---> SHRUBOUS -> (A*D(ASDHJAIP**(@#$(@#$
(A*D(ASDHJAIP**(@#$(@#$ ---> SHRUBOUS -> INFORMAÇÃO
```

Chave pública e privada (criptografia assimétrica)

A criptografia de chave pública ou criptografia assimétrica é um método de criptografia que utiliza um par de chaves: uma chave pública e uma chave privada. A chave pública é distribuída livremente para todos os correspondentes via e-mail ou outras formas, enquanto a chave privada deve ser conhecida apenas pelo seu dono.

Num algoritmo de criptografia assimétrica, uma mensagem cifrada (encriptada é um termo incorrecto) com a chave pública pode somente ser decifrada pela sua chave privada correspondente.

Os algoritmos de chave pública podem ser utilizados para autenticidade e confidencialidade. Para confidencialidade, a chave pública é usada para cifrar mensagens, com isso apenas o dono da chave privada pode decifrá-la. Para autenticidade, a chave privada é usada para cifrar mensagens, com isso garante-se que apenas o dono da chave privada poderia ter cifrado a mensagem que foi decifrada com a 'chave pública'.

```
INFORMACAO --> CHAVE PRIVADA --> &*!!YV#OU!F#O!&*@#(
&*!!YV#OU!F#O!&*@#( --> CHAVE PRIVADA --> sidtfysoudftsdouf
&*!!YV#OU!F#O!&*@#( --> CHAVE PUBLICA --> INFORMACAO
INFO --> CHAVE PUBLICA --> @#$(*@#$(SJDASD
@#$(*@#$(SJDASD --> CHAVE PRIVADA --> INFO
```

SSL

O SSL, ou Secure Sockets Layer, é um padrão Web que permite trafegar dados sensíveis e confidenciais com segurança através da internet.

O protocolo HTTPS é utilizado em bancos e empresas que utilizam autenticação com criptografia depende da configuração do SSL, bem como a criação de chaves e certificados.

O Apache que trabalha com SSL usando o módulo mod_ssl.

O primeiro passo é instalar o pacote openssl, caso ainda não esteja instalado:

```
# aptitude install openssl
```

Linux Network Servers

Verifique se o módulo SSL está habilitado, e, em caso negativo, habilite-o:

```
# a2enmod ssl
```

O protocolo HTTPS trabalha na porta 443, então, é necessário fazer com que o apache ouça nesta porta:

```
# echo "Listen 443" >> /etc/apache2/ports.conf
```

Certificado digital:

Um certificado digital é um arquivo de computador que contém um conjunto de informações referentes a entidade para o qual o certificado foi emitido (seja uma empresa, pessoa física ou computador) mais a chave pública referente a chave privada que acredita-se ser de posse unicamente da entidade especificada no certificado.

Um certificado digital normalmente é usado para ligar uma entidade a uma chave pública.

Para garantir digitalmente, no caso de uma Infraestrutura de Chaves Públicas (ICP), o certificado é assinado pela Autoridade Certificadora que o emitiu e no caso de um modelo de Teia de Confiança (Web of trust) como o PGP, o certificado é assinado pela própria entidade e assinado por outros que dizem confiar naquela entidade.

Em ambos os casos as assinaturas contidas em um certificado são atestamentos feitos por uma entidade que diz confiar nos dados contidos naquele certificado.

O SSL trabalha com o conceito de certificados públicos, então, devemos efetuar os procedimentos de criação do certificado que será fornecido aos clientes.

Crie a chave que será usada para assinar o certificado:

```
# openssl genrsa -out /etc/ssl/microX.key 1024
```

Linux Network Servers

Com a chave em mãos, crie o certificado (fique atento as perguntas)

```
# openssl req -new -key /etc/ssl/microX.key -out /etc/ssl/microX.csr
```

Depois de criar o certificado, você pode enviá-lo a uma unidade certificadora, que o assinará por um valor anual, ou, caso você mesmo pode assinar o certificado, lembrando que, neste caso, o cliente dirá que o certificado não foi reconhecido por uma unidade certificadora.

```
# openssl x509 -req -days 365 -in /etc/ssl/microX.csr -signkey /etc/ssl/microX.key -out /etc/ssl/microX.crt
```

Após gerar o certificado, configure seu domínio Virtual:

```
NameVirtualHost *:443
<VirtualHost *:443>
DocumentRoot /var/www/seunome.com.br
ServerName *:443
ServerAdmin webmaster@seunome.com.br
ErrorLog /var/log/apache2/seunome.com.br-error.log
CustomLog /var/log/apache2/seunome.com.br-access.log common
SSLEngine on
SSLCertificateFile /etc/ssl/microX.crt
SSLCertificateKeyFile /etc/ssl/microX.key
</VirtualHost>
NameVirtualHost www.seunome.com.br:80
<VirtualHost www.seunome.com.br:80>
DocumentRoot /var/www/seunome.com.br
</VirtualHost>
```

Linux Network Servers

O arquivo .htaccess funciona como uma extensão de um Virtual Host, onde é possível escrever configurações especificas sem a necessidade de gerar uma configuração de inteira de Virtual Host.

Aqui, nós utilizaremos este arquivo para escrever uma regra de redirecionamento para forçar o uso de HTTPS no nosso site.

Crie um arquivo chamado .htaccess dentro do diretório /var/www/seunome.com.br, e coloque o seguinte conteúdo:

```
RewriteEngine On
RewriteCond %{SERVER_PORT} 80
RewriteRule ^(.*)$ https://www.seunome.com.br/$1 [R,L]
```

Ative o mod_rewrite para que, toda vez que alguém acessar seu site, seja automaticamente redirecionado para o site com HTTPS ativado:

```
# a2enmod rewrite
```

A função dele é reescrever URL a partir de um conjunto de parâmetros.

Reinicie o apache:

```
# invoke-rc.d apache2 restart
```

Agora, abra o navegador e efetue o teste no endereço:

http://www.seunome.com.br

Se quiser aumentar o nível de segurança de sua empresa, utilize certificados válidos assinados por uma unidade certificadora!

Exemplo de unidades certificadoras famosas:

Verisign, Thawte, Geotrust, Network Solutions.

Linux Network Servers

Visto o exemplo acima, vamos gerar um certificado self-signed para o nosso site teste-ht.

Vamos utilizar um script chamado "make-ssl-cert"

Para isso temos que instalar o seguinte pacote:

```
# aptitude install ssl-cert
# mkdir /etc/apache2/ssl
# cd /etc/apache2/ssl
# make-ssl-cert /usr/share/ssl-cert/ssleay.cnf apache.pem -days 365
Country Name (2 letter code) [AU]:BR
State or Province Name (full name) [Some-State]:SP
Locality Name (eg, city) []:Sao Paulo
Organization Name (eg, company) [Internet Widgits Pty Ltd]:HackerTeen
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:www.teste-ht.com.br
Email Address []:root@teste-ht.com.br
-days define a validade do certificado
1 ano = 365 dias
```

Devemos alterar o seguinte arquivo:

```
# vim /etc/apache2/sites-available/teste-ht

NameVirtualHost * -> NameVirtualHost *:443
<VirtualHost *> -> <VirtualHost *:443>
```

* Acrescentar dentro de <Virtualhost *:443>

Linux Network Servers

```
SSLEngine On
ServerSignature On
SSLCertificateFile /etc/apache2/apache.pem
```

* Configurar a porta

```
vim /etc/apache2/ports.conf
```

Acrescentar uma linha com: Listen 443

```
apache2ctl -S
```

* Ativar o modulo ssl

```
a2enmod ssl
```

* Reinicia o apache:

```
/etc/init.d/apache2 restart
```

Acessar https://www.teste-ht.com.br.

* Verificar se está funcionando:

```
netstat -ptln | grep 443
```

O resultado final é esse:

```
NameVirtualHost *:443
<VirtualHost *:443>
SSLEngine On
ServerSignature On
```

Linux Network Servers

```
SSLCertificateFile /etc/apache2/apache.pem
DocumentRoot /srv/teste-ht
ServerName www.teste-ht.com.br
ServerAdmin webmaster@teste-ht.com.br
ErrorLog /var/log/apache2/teste-ht-error.log
CustomLog /var/log/apache2/teste-ht-access.log common
</VirtualHost>
```

E:

```
Listen 80
Listen 443
```

No ports.conf

Módulo para controle de banda

```
mod_cband
```

Ele permite limitar o uso de banda ou número de conexões simultâneas atendidas pelos sites que são hospedados pelo o Apache.

Nas distribuições derivadas do Debian, ele pode ser instalado através do pacote "libapache2-mod-cband", como em:

```
# aptitude install libapache2-mod-cband
```

Com o módulo instalado, use o comando a2enmod para ativá-lo:

```
# a2enmod cband
```

Em distribuições derivadas do Red Hat é necessário instalar o pacote a partir do código fonte.

Linux Network Servers

Instale o pacote "httpd-devel" usando o yum:

```
# yum install httpd-devel
```

Em seguida, baixe a versão mais recente do pacote no: http://sourceforge.net/projects/cband/

```
Como instalar:

# tar -zxvf mod-cband-0.9.6.1.tgz
# cd mod-cband-0.9.6.1
# ./configure
# make
# make install
```

O script de instalação se encarrega de instalar o módulo na pasta "/usr/lib/httpd/modules/" e adicionar uma linha similar à linha a seguir no arquivo "/etc/httpd/conf/httpd.conf" (Isso em distribuições baseadas no Red Hat), de forma que ele seja carregado:

```
LoadModule cband_module /usr/lib/httpd/modules/mod_cband.so
```

Falta agora apenas editar a configuração de cada virtual host, definindo os limites desejados, como em:

```
NameVirtualHost *
<VirtualHost *>
DocumentRoot /srv/teste-ht
ServerName www.teste-ht.com.br
ServerAlias teste-ht.com.br
```

Linux Network Servers

```
ErrorLog /var/log/apache2/teste-ht-error.log
CustomLog /var/log/apache2/teste-ht-access.log common
CBandSpeed 1024kbps 10 20
CBandRemoteSpeed 512kbps 5 3
</VirtualHost>
```

A opção "CBandSpeed" determina os limites para o tráfego total do site, especificando o volume máximo de banda que pode ser usado, o número máximo de requisições de páginas e/ou arquivos por segundo e o número máximo de instâncias do Apache que podem ser utilizadas.

No exemplo, limitei o tráfego do site a 1024 kbps, com um máximo de 10 requisições por segundo e um máximo de 20 conexões simultâneas.

Em seguida, temos a opção "CBandremoteSpeed", que permite definir limites individuais para os visitantes, impedindo que um único usuário monopolize toda a banda disponível (pense no caso de um site que disponibiliza arquivos para download, por exemplo).

No exemplo, cada usuário ficará limitado a um máximo de 256 kbps, com até 3 requisições por segundo e um máximo de três conexões simultâneas.

É necessário reiniciar o Apache para que as alterações sejam aplicadas:

```
# /etc/init.d/apache2 force-reload
```

Para definir a quota de tráfego, o primeiro passo é criar uma pasta onde o cband armazenará informações sobre o tráfego usado por cada host.

Você pode tanto criar uma pasta dentro do diretório do site e restringir o acesso a ela quanto criar uma pasta em outro diretório do sistema.

Linux Network Servers

O mais importante é que você ajuste as permissões de acesso, de forma que o dono seja o usuário usado pelo Apache ("www-data" no Debian), como em:

```
# mkdir /var/www/scoreboard
# chown www-data:www-data /var/www/scoreboard/
```

Dentro desse diretório, criamos um arquivo vazio para cada site onde formos ativar o cband, novamente dando a posse para o usuário usado pelo Apache, como em:

```
# touch /var/www/scoreboard/teste-ht
# chown www-data:www-data /var/www/scoreboard/teste-ht
```

Em seguida, precisamos alterar a configuração dos sites (dentro da pasta "/etc/apache2/sites-available"), especificando a nova configuração, como em:

```
<VirtualHost *:80>
DocumentRoot /srv/teste-ht
ServerName www.teste-ht.com.br
ServerAlias teste-ht.com.br
ErrorLog /var/log/apache2/teste-ht-error.log
CustomLog /var/log/apache2/teste-ht-access.log common
CBandLimit 100G
CBandPeriod 4W
CBandScoreboard /var/www/scoreboard/teste-ht
</VirtualHost>
```

Como você pode imaginar, a opção "CBandLimit" especifica a quota de tráfego do site, que no exemplo foi definida como 100 GB (além do "G" você pode especificar o volume em "M", de megabytes ou "K", de kbytes).

A linha seguinte, "CBandPeriod", determina o período de aplicação da quota, depois do qual a contagem é zerada. No exemplo usei "4W", que especifica um

Linux Network Servers

período de 4 semanas, ou seja, 28 dias. Você poderia usar outros valores, como por exemplo "30D" (30 dias) ou "24H" (24 horas).

Em seguida, temos a opção "CBandScoreboard", que indica o arquivo onde serão armazenadas as informações sobre o uso de banda do site (que criamos no passo anterior).

Depois de cada alteração, não se esqueça de atualizar a configuração do Apache para que ela entre em vigor:

```
# /etc/init.d/apache2 reload
```